الاسـبوع
الثامـــن

أنظمة الملفـات
(File Systems)

الملفــات (Files) :

يتضمن هذا الموضوع وجهة نظر المستخدم .

تسمية الملف (File name) :

عندما تنشىء عملية ما ملفاً ، فإنها تعطيه اسماً. عندما تنتهي العملية يبقى الملف موجوداً ويمكن الوصول إليه من قبل عملية أخرى باستخدام اسمه .

تختلف تفاصيل قواعد تسمية الملفات من نظام إلى آخر ، لكن جميع أنظمة التشغيل الحالية تسمح بتسمية الملفات بسلاسل من حرف إلى ثمانية حروف . وبالتالي ، تعتبر " احمد " و " المعهد " و " computer" و " fr " أسماء ملفات صالحـة ، وغالبا تكون الأرقام والرموز الخاصة مسموحة ، لذلك تعتبر الأسماء "2 " و " احمد4 " و "computer!" مقبولة غالبا . تدعم العديد من أنظمة التشغيل أسماء طويلة حتى (255 حرف ورمز ورقم) .

تفرق بعض أنظمة التشغيل بين الأحرف الكبيرة والصغيرة ، بينما لا تميز بينهما أنظمة أخرى . يصنف (UNIX) في المجموعة الأولى بينما يصنف (MS-DOS) في الثانية . لذلك يمكن في (UNIX) وجود ثلاثة ملفات بالأسماء maria و Maria و MARIA . أما في (MS-DOS) فتشير جميع هذه الأسماء إلى نفس الملف.

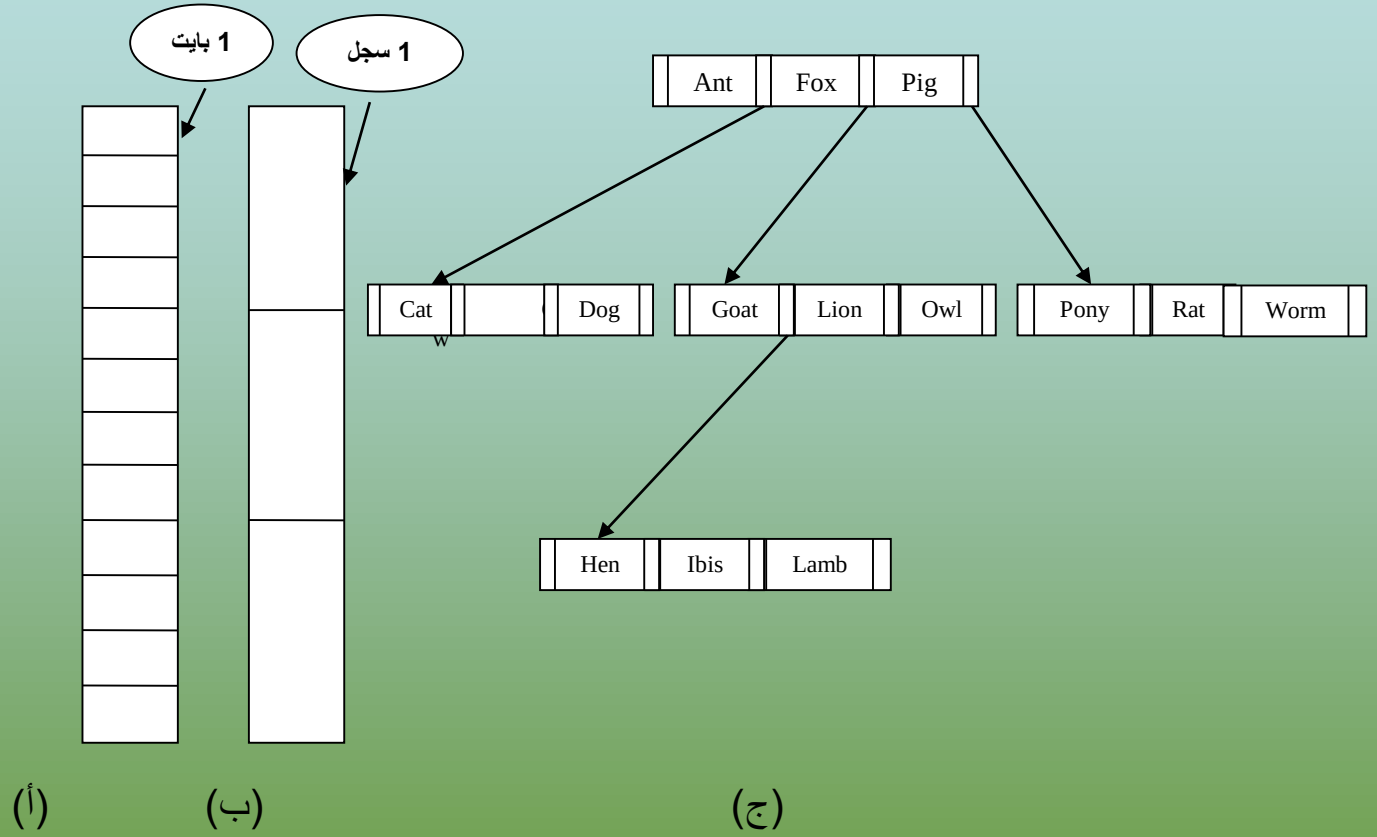
تدعم العديد من أنظمة التشغيل أسماء ملفات ذات جزأين ، حيث تفصل ما بيـن الجزأين نقطة كمـا في computer.c . يسمى الجزء الذي يلي النقطة بامتداد الملف (File Extension) وعادة ما يدل على خاصية معينة للملف . مثلا ، يتألف اسم الملف في (MS-DOS) من 1 إلى 8 حرف أو رمز أو رقم بالإضافة إلى امتداد اختياري من 1 إلى 3 حرف أو رمز أو رقم . وقد يمتلك الملف امتدادين أو أكثر كما في (UNIX) فمثلا، prog.c.z حيث تستخدم z للدلالة على أن الملف قد ضغط باستخدام خوارزمية (zip) للضغط .

الجدول (1) يبين بعض امتدادات الملفات الأكثر شهرة ومعانيها

الامتداد	نوع الملف
.dll	System library
.Bsc	Basic file
.Pas	Pascal file
.c	C file
.cpp	C++ file
.doc	Word file
.exe	Executable file
.txt tex	Text file

هيكلية الملف (File Structure) :

يمكن بناء الملفات بعدة أساليب مختلفة ، ثلاث أساليب شائعة الاستخدام . المبين في الشكل (12أ) عبارة عن تسلسل غير مهيكل من البايتات . في الواقع ، لا يعرف نظام التشغيل ولا يهتم بمحتويات الملف . كل ما يراه هو مجموعة بايتات . يتم تحديد المعنى من قبل برامج المستخدم ، يستخدم كل من Unix و Windows هذا الأسلوب .



الشكل (12) ثلاثة أنواع من الملفات. أ) تسلسل بايتات. ب) تسلسل سجلات. ج) شجرة.

إن جعل نظام التشغيل يرى الملفات مجرد تسلسل بايتات لا أكثر يعطي المرونة الأكبر. تستطيع برامج المستخدم وضع أي شيء تريده في ملفاتها وتسميها بأي طريقة تلائمها.

يبين الشكل (12ب) نموذجا أكثر هيكلية. يتألف الملف في هذا النموذج من سلسلة من السجلات الثابتة الحجم ، كل منها لها بنية داخلية معينة . بما أن الملف مكون من تسلسل من السجلات ، فان كل عملية قراءة تقرأ سجلا واحدا وكل عملية كتابة تكتب أو تضيف سجلا واحدا. لا يوجد نظام عام الأغراض يعمل بهذا الأسلوب حاليا.

يبين الشكل (12ج) النوع الثالث من هيكلية الملفات . يتألف الملف من شجرة سجلات، لا تكون سجلاتها متساوية الحجم بالضرورة ، يحوي كل سجل حقل مفتاح (Key) موجود في مكان ثابت من السجل. ترتب الشجرة حسب حقل المفتاح للسماح بالبحث السريع عن مفتاح معين . يمكن الحصول على السجل من خلال مفتاحه دون الاهتمام بموقعه الصحيح في الملف. أضف إلى ذلك، يمكن إضافة سجلات جديدة حين يقرر نظام التشغيل .

أنواع الملفات (File Types):

العديد من أنظمة التشغيل تدعم أنواعا متعددة من الملفات . مثلا ، يملك UNIX و Windows ملفات نظامية وأدلة . يوجد في UNIX أيضا ملفات خاصة . الملفات النظامية (Regular Files) هي الملفات التي تحوي معلومات المستخدم . جميع الملفات في الشكل (12) أعلاه هي ملفات نظامية . الأدلة (Directories) هي ملفات نظام تقوم بتنظيم هيكلة وبنية نظام الملفات . سنتطرق عليها بعد قليل .

الملفات النظامية بشكل عام إما أن تكون ملفات ASCII أو ملفات ثنائية. تتألف ملفات ASCII من أسطر من النص. ينتهي كل سطر في بعض الأنظمة برمز الرجوع (carriage return) ، وتستخدم بعض الأنظمة مثل (MS-DOS) نفس الرمز أعلاه أو رموز أخرى . ليس من الضروري أن تكون الأسطر متساوية الطول.

الميزة الكبرى لملفات ASCII انه يمكن عرضها وطباعتها كما هي ، ويمكن تحريرها بأي محرر نصوص. أما الملفات الأخرى هي ملفات ثنائية ، وهذا يعني أنها ليست ملفات ASCII . تعطي طباعة هذه الملفات على الطابعة إخراج غير مفهوم ومليء بالرموز الغريبة والعشوائية . عادة ، تكون البنية الداخلية لهذه الملفات معروفة فقط للبرامج التي تستخدمها.

طرائق الوصول للملفات (Files Access):

1- الوصول التسلسلي (Sequential Access): قدمت أنظمة التشغيل الأولى نوعا واحدا فقط من الوصول إلى الملفات ، تستطيع العملية في هذه الأنظمة قراءة جميع البايتات أو السجلات في الملف بالترتيب ، ابتدءا من بداية الملف ، لكنها لا تستطيع اجتياز بعضها وقراءتها بترتيب مختلف. إلا انه من الممكن إعادة لف الملف التسلسلي إلى البداية وقراءته مرات ومرات حسب الحاجة .

كانت الملفات التسلسلية مناسبة عندما كان وسط التخزين عبارة عن أشرطة مغناطيسية وليس أقراصا . ولكن بعد ظهور الأقراص ظهرت مشكلة الوصول للملفات .

2- الوصول العشوائي (Random Access): عندما بدأ استخدام الأقراص لتخزين الملفات، أصبح من الممكن قراءة البايتات أو السجلات من الملف دون ترتيب ، أو الوصول إلى سجل بواسطة المفتاح ، عوضا عن الموقع. تسمى الملفات التي يمكن قراءة بايتاتها أو سجلاتها بأي ترتيب بملفات الوصول العشوائي، وهي مطلوبة من قبل العديد من البرامج التطبيقية.

هناك طريقتان لتحديد مكان بدء القراءة. تعتمد الطريقة الأولى على إعطاء كل عملية read المكان الذي يراد القراءة منه. أما الطريقة الثانية فتعتمد على عملية خاصة اسمها seek مهمتها تحديد الموقع الحالي. بعد إجراء seek يمكن قراءة الملف بشكل تسلسلي ابتدءا من الموقع الحالي الجديد .

مواصفات الملفات (File Attributes):

كل ملف له اسم ويحوي بيانات. بالإضافة إلى ذلك ، تربط جميع أنظمة التشغيل معلومات أخرى بكل ملف، مثل تاريخ ووقت إنشاء الملف وحجم الملف . تسمى هذه المعلومات الإضافية بسمات ومواصفات الملف (Attributes) . تختلف لائحة المواصفات بشكل كبير من نظام تشغيل إلى آخر. يبين الجدول (2) بعض المواصفات الممكنة ، ولكن هناك مواصفات أخرى موجودة أيضا ، لا يوجد أي نظام حالي يملك جميع هذه المواصفات ، لكن كلا منها يملك بعضها .

جدول (2) بعض مواصفات الملفات الممكنة

المواصفات	المعنـــــى
الحماية	تحدد من يستطيع الوصول إلى الملف وكيف يتم ذلك.
كلمة المرور	كلمة المرور اللازمة للوصول إلى الملف
المنشىء	رقم تعريف (ID) الشخص الذي أنشأ الملف.
المالك	المالك الحالي للملف.
علم القراءة فقط	0 للقراءة والكتابة ، 1 للقراءة فقط .
علم الإخفاء	0 ملف عادي ، 1 لا يظهر عند سرد الملفات .
علم النظام	0 ملف عادي ، 1 ملف خاص بنظام التشغيل .
علم الأرشفة	0 تم نسخه احتياطيا ، 1 يحتاج للنسخ الاحتياطي .
علم ثنائي أو ASCII	0 ملف ASCII ، 1 ملف ثنائي .
علم الوصول العشوائي	0 وصول تسلسلي ، 1 وصول عشوائي .
علم الملف المؤقت	0 ملف عادي ، 1 حذف الملف بعد انتهاء العملية .
أعلام القفل	0 غير مقفول ، غير الصفر ملف مقفول .
طول السجل	عدد البايتات في السجل.
موقع المفتاح	إزاحة المفتاح في كل سجل.
زمن الإنشاء	وقت وتاريخ إنشاء الملف.
زمن الوصول الأخير	وقت وتأريخ الوصول الأخير للملف.
زمن التعديل الأخير	وقت وتأريخ التعديل الأخير للملف.
الحجم الحالي	عدد البايتات في الملف.
الحجم الأعظمي	عدد البايتات التي يمكن وضعها في الملف.

تتعلق المواصفات الأربع الأولى بحماية الملف وتحديد من يمكنه الوصول إليه ومن لا يمكنه ذلك . في بعض الأنظمة يجب على المستخدم أن يقدم كلمة مرور للوصول إلى الملف ، وفي هذه الحالة يجب أن تكون كلمة المرور من مواصفات الملف .

الأعلام عبارة عن بتات أو حقول قصيرة تتحكم أو تفعّل بعض الخصائص المعينة . الملفات المخفية مثلا لا تظهر عند سرد جميع الملفات .

العمليات الممكن تنفيذها على الملفات (Files Operations)

أوجدت الملفات لتخزين المعلومات والسماح باسترجاعها لاحقاً . تقدم الأنظمة المختلفة عمليات مختلفة للسماح بالتخزين والاسترجاع ، ومن العمليات الأكثر شيوعاً المتعلقة بالملفات :

1- Create : إنشاء ملف بدون بيانات . الغرض من هذا الاستدعاء الإعلان عن وجود الملف وتحديد بعض مواصفاته .

2- Delete : عندما يصبح الملف لا حاجة لوجوده ، يجب حذفه لتحرير مساحة القرص. يوجد دائماً استدعاء نظام من أجل هذا الغرض.

3- Open : قبل استخدام ملف ، يجب على العملية أن تفتحه. الغرض من الاستدعاء open السماح للنظام بإحضار مواصفات الملف ولائحة بعناوين القرص إلى الذاكرة من اجل الوصول السريع في الاستدعاءات اللاحقة.

4- Close : عند الانتهاء من جميع عمليات الوصول، لا تعود مواصفات الملف ولائحة عناوين القرص لازمة، لذلك يجب إغلاق الملف لتحرير مساحة الجدول الداخلي المحجوزة من اجله.

5- Read : قراءة البيانات من الملف. تأتي البيانات عادةً من الموقع الحالي . يجب على المستدعي أن يحدد كمية البيانات التي يحتاجها ويجب أن يقدم المخزن المؤقت الذي ستوضع البيانات فيه.

6- Write : كتابة بيانات إلى ملف. تكتب البيانات أيضاً في الموقع الحالي . إذا كان الموقع الحالي هو نهاية الملف ، يزداد حجم الملف. إذا كان الموقع الحالي في وسط الملف، تتم الكتابة فوق البيانات القديمة والتي تضيع إلى الأبد .

7- Append : هذا الاستدعاء هو شكل مقيد من Write . ويستطيع الكتابة فقط في نهاية الملف.

8- Seek : تحتاج الملفات ذات الوصول العشوائي إلى طريقة لتحديد المكان الذي ستقرأ منها البيانات .من الأساليب الشائعة وجود استدعاء نظام اسمه seek يحرك مؤشر الملف إلى مكان معين في الملف. بعد هذا الاستدعاء يمكن قراءة البيانات من هذا الموقع أو كتابتها إليه.

9- Get attributes : تحتاج العمليات غالبا لقراءة مواصفات وسمات ملف للقيام بعملها. مثلاً ، يستخدم البرنامج make في نظام UNIX لإدارة تطوير البرمجيات المؤلفة من عدة ملفات مصدرية . عندما يستدعى make ، يفحص أزمان تعديل جميع الملفات المصدرية والملفات الموافقة لها لإجراء أقل عدد ممكن من عمليات الترجمة .

10- Set attributes : بعض المواصفات يمكن تحديدها من قبل المستخدم ويمكن تغييرها بعد إنشاء الملف. يسمح استدعاء النظام هذا بالقيام بهذه العملية .

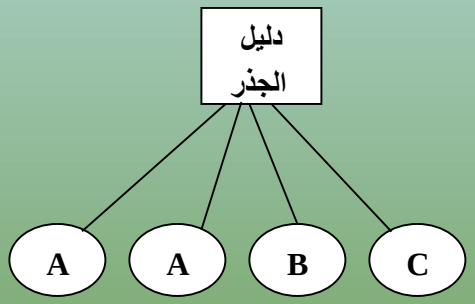
11- Rename : من الشائع أن يحتاج مستخدم لأن يغير اسم ملف موجود. يسمح هذا الاستدعاء بذلك. هذه العملية ليست ضرورية دائماً، لأنَه من الممكن نسخ الملف إلى ملف جديد بالاسم الجديد ثم مسح الملف القديم.

الأدلة والمجلدات (Directories) :

لتنظيم الملفات، تحوي أنظمة الملفات عادةً أدلة (Directories) أو مجلدات (Folders) ، والتي تكون في معظم الأنظمة عبارة عن ملفات أيضاً.

الأدلة ذات المستوى الواحد (Single Level Directories) :

الشكل الأبسط لأنظمة الأدلة هو وجود دليل واحد يحوي جميع الملفات . يدعى هذا الدليل أحياناً بدليل الجذر (Root Directory) ، لكن باعتبار أنه الدليل الوحيد ، فإن الاسم لا يهم كثيراً . كان شائعا في أنظمة الحواسب الشخصية الأولى ، وذلك لأنه لا يوجد إلاّ مستخدم واحد فقط. يبين الشكل (13) مثالا عن نظام ذي دليل واحد . يحوي الدليل هنا أربعة ملفات . يبين الشكل مالكي الملفات وليس أسماؤها . تتمثل محاسن هذا النموذج في بساطته وقدرته على إيجاد الملفات بسرعة لأنه يوجد فقط مكان واحد للبحث فيه.

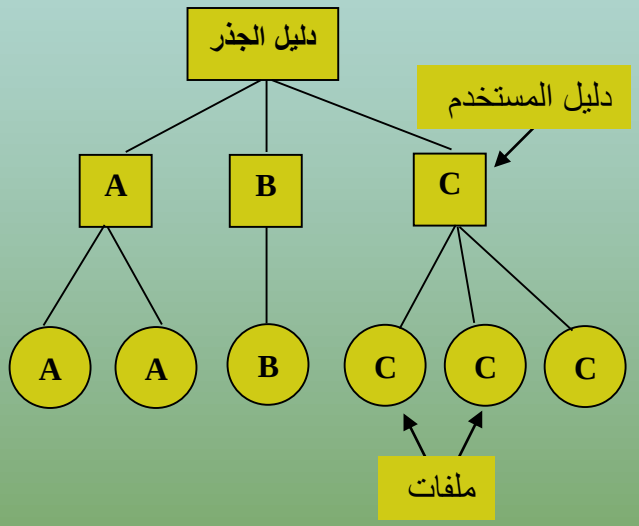


الشكل(13) نظام دليل ذو مستوى واحد يحوي أربعة ملفات، يملكها ثلاثة أشخاص A و B و C

المشكلة في وجود دليل واحد فقط في نظام متعدد المستخدمين إن المستخدمين المختلفين قد يستخدمون بالصدفة نفس الأسماء لملفاتهم . فمثلا ، إذا أنشأ المستخدم A ملفا اسمه (mailbox) ثم أنشأ المستخدم B بعد ذلك ملفا اسمه (mailbox) أيضاً، فان ملف المستخدم B سيكتب فوق ملف المستخدم A . نتيجة لذلك ، لم تعد هذه الطريقة مستخدمة في الأنظمة متعددة المستخدمين.

الأدلة ذات المستويين (Two Level Directories):

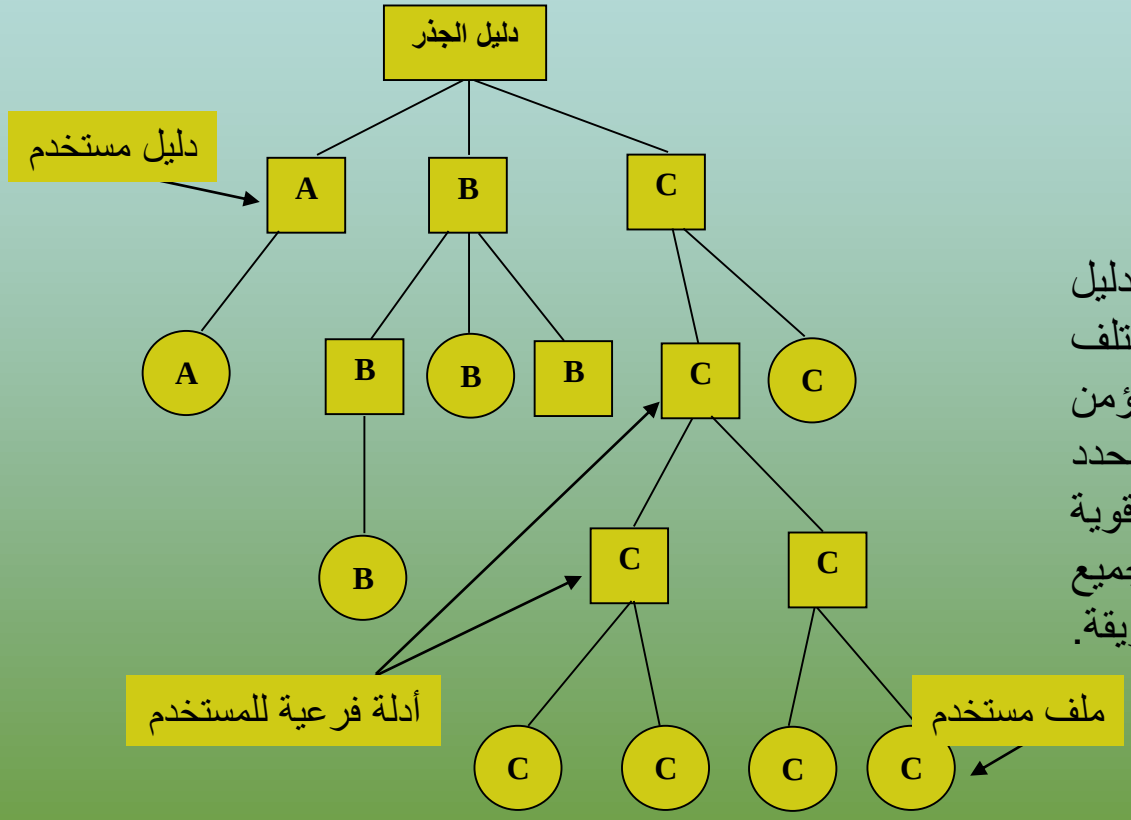
لتجنب التعارضات الناتجة عن تسمية مستخدمين مختلفين ملفاتهم بنفس الاسم ، يعطي كل مستخدم دليلاً خاصاً به. بهذه الطريقة ، لا تتداخل الأسماء التي يعطيها احد المستخدمين لملفاته مع أسماء ملفات مستخدم آخر ولا يوجد مشكلة في وجود نفس اسم الملف في دليلين مختلفين أو أكثر. يبين الشكل (14) هذا التصميم . يمكن استخدام هذا التصميم مثلا في حاسوب متعدد المستخدمين أو في شبكة صغيرة من الحواسب الشخصية التي تتشارك بمخدم ملفات مشترك عبر شبكة محلية .



الشكل (14) دليل ذات مستويين تدل الحروف على مالكي الملفات والأدلة

الأدلة ذات المستويات التدرجية (Hierarchical Level Directories):

يتخلص نظام الأدلة ذات المستويين من تعارض الأسماء بين المستخدمين لكنه ليس مرضيا للمستخدمين الذين يملكون عددا كبيرا من الملفات. من الشائع أن يرغب المستخدمون بتجميع ملفاتهم ضمن مجموعات منطقية . إن ما نحتاجه هو نظام هرمي تدرجي عام (أي شجرة من الأدلة) . بهذه الطريقة، يستطيع كل مستخدم إنشاء أي عدد يريده من الأدلة بحيث يمكنه تجميع الملفات بطريقة طبيعية . يبين الشكل (15) هذا الأسلوب.



الأدلة A و B و C موجودة في دليل الجذر وينتمي كل منها إلى مستخدم مختلف ، اثنان منهما أنشأ أدلة فرعية ، تؤمن إمكانية إنشاء المستخدمين عددا غير محدد من الأدلة الفرعية أداة هيكلية قوية للمستخدمين لتنظيم عملهم. تنظم جميع أنظمة الملفات الحديثة تقريبا بهذه الطريقة.

الشكل (15) نظام أدلة تدرجي (هرمي)

تسمية الممر الموصل للدليل (Path Names):

عندما يُنظَّم نظام الملفات على شكل شجرة من الأدلة ، نحتاج إلى طريقة ما لتحديد أسماء الملفات. هناك طريقتان شائعتا الاستخدام :

1- المسار المطلق (Absolute Path Name):

يتم إعطاء اسم مسار يتألف ابتداءاً من دليل الجذر حتى الملف. كمثال عن ذلك نأخذ المسار /usr/ast/mailbox ويعني أن دليل الجذر يحوي دليلاً فرعياً اسمه usr والذي يحوي بدوره الدليل الفرعي ast والذي يحوي الملف mailbox . تبدأ أسماء المسارات دائماً بدليل الجذر وتكون فريدة. تُفصل مكونات المسار في نظام UNIX بالرمز (/) ، أما في Windowsفيستخدم الرمز (\ ) وفي نظام التشغيل Ms-Dosالرمز (\ ) أيضاً ونظام MULTICS يستخدم الرمز (<) . وبالتالي يُكتب نفس اسم المسار في الأنظمة الثلاثة كما يلي :

Windows	\usr\ast\mailbox
UNIX	/usr/ast/mailbox
MULTICS	>usr>ast>mailbox

بغض النظر عن الرمز المستخدم إذا كان الحرف الأول هو الرمز الفاصل فهذا يعني أن المسار مطلق.

2- المسار النسبي (Relative Path Name) :

تستخدم هذه الأسماء بالارتباط مع مفهوم الدليل الحالي (Current Directory) . يستطيع المستخدم أن يحدد دليلاً على انه الدليل الحالي ، وفي هذه الحالة تنسب جميع المسارات التي لا تبدأ بدليل الجذر إلى الدليل الحالي . مثلاً ، إذا كان الدليل الحالي هو /usr/ast ، فإن الملف الذي مساره المطلق هو /usr/ast/mailbox يمكن الإشارة إليه ببساطة بالاسم mailbox . بعبارة أخرى، يعتبر أمر UNIX التالي:

```
cp    /usr/ast/mailbox          /usr/ast/mailbox.bak
```

والأمر :

```
cp     mailbox      mailbox.bak
```

متكافئين تماماً إذا كان الدليل الحالي هو/usr/ast . يعتبر المسار النسبي عادةً أنسب ، لأنه أقصر ويقوم بنفس عمل المسار المطلق .

عمليات الأدلة (Directory Operations):

1- Create إنشاء دليل . ويكون فارغاً إلاّ من النقطة والنقطتين ، حيث توضعان تلقائياً من قبل النظــام (وفي حالات أخرى من قبل البرنامج mkdir).

2- Delete حذف دليل . يمكن حذف الأدلة الفارغة فقط. يعتبر الدليل الذي يحوي النقطة والنقطتين فقط دليلاً فارغاً حيث لا يمكن حذف النقاط.

3- Opendir التمكن من قراءة دليل، مثلاً ، لعرض جميع الملفات في دليل ، يقوم البرنامج بفتح الدليل لقراءة أسماء جميع الملفات التي يحتويها. قبل التمكن من قراءة دليل ، يجب أن يفتح أولاً، وهذا يشبه فتح الملف وقراءته.

4- Closedir بعد قراءة دليل ، يجب أن يغلق لتحرير المساحة المحجوزة له .

5- Readdir يعيد هذا الاستدعاء العنصر التالي في دليل مفتوح . سابقاً، كان من الممكن قراءة الأدلة باستخدام استدعاء النظام read ، لكن هذه الطريقة تعاني من سيئة تتمثل في اضطرار المبرمج لتعلم البنية الداخلية للأدلة والتعامل معها. على العكس من ذلك، يعيد readdir دائماً عنصراً واحداً . بغض النظر عن بنى الأدلة المتعددة الممكن استخدامها.

6- Rename تشبه الأدلة الملفات في كثير من النواحي ويمكن إعادة تسمية الأدلة بنفس طريقة الملفات تماماً.

7- Link الربط عبارة عن تقنية تسمح بظهور الملف في أكثر من دليل واحد. يحدد استدعاء النظام هذا ملفاً موجوداً وأسم مسار، وينشىء وصلة بين الملف والمسار المحدد. بهذه الطريقة، يظهر نفس الملف في أدلة متعددة. يسمى هذا النـوع من الوصـلات التي تتبَـع عدد ملفات الأدلـة التي تشير إلى نفس الملف باسم الوصلة الصلبة (Hard Link) .

8- Unlink إزالة ملف من دليل. إذا كان الملف الذي سيزال ارتباطه موجوداً في دليل واحد فقط (الحالة العادية) فإنه يحذف من نظام الملفات. أما إذا كان موجودا في عدة أدلة ، فإن اسم المسار المحدد هو الذي سيزال، وتبقى الأسماء الباقية.

ملاحظة:

الاستدعاءات أعلاه مأخوذة من نظام UNIX وتختلف الأنظمة فيما بينها في هذه الاستدعاءات.

اسئلة اختبارية:

س1: لماذا نظام التشغيل DOS لا يميز بين اسماء الملفات التالية : class , CLass , CLASS

س2: ماذا يعني وجود اكثر من امتداد في تسمية الملفات في نظام Unix ؟ مع اعطاء مثال لاسم ملف في النظام اعلاه .

س3: ما هو الفرق في هيكلية الملف بين اسلوب (تسلسل بايتات) واسلوب (تسلسل سجلات) ؟

س4: عدد مستويات الادلة ؟

س5: ما هي انواع المسارات في نظام الملفات وبشكل عام لجميع نظم التشغيل معززا الاجابة بامثلة ؟